Solution to HW5

AP5.2 We are given a feedback system whose forward path has three poles and no or one zero. We are asked to find the percent overshoots, rise times, settling times, zeros, and closed-loop poles for values 0, 0.05, 0.1, and 0.5 of the parameter $\tau_z$. The parameter $\tau_z$ determines the location of the zero. (When $\tau_z = 0$, the system has no zero.)

Solution: The following matlab script achieves these objectives. The transcript and the plot of the step responses are also shown. Note also that for $\tau_z = 0.1$, the computed percentage overshoot is negative. In this case, there was no overshoot and so the percentage overshoot is undefined. The value is an artifact of the way I wrote the script. From the plot we see that the system response improves (overshoot decreases, rise time decreases) initially as $\tau_z$ increases from 0 to 0.1, but then worsens (overshoot increases and oscillations increase) for $\tau_z = 0.5$.

Although the real part of the dominant roots is not less than 1/10 the real part of the third root (the criterion for approximating the system using the dominant roots, given on page 234), solving for the damping coefficient corresponding to the pair of complex conjugate roots as if we could neglect the real root does give some insight into the system behavior. Taking the roots of $s^2 + 2\omega_n \omega_s + \omega_n^2 = 0$ yields $s = \sigma \pm j\omega = -\omega_n \pm j\omega_n \sqrt{1 - \zeta^2}$. Solving for $\zeta$ in terms of $\sigma$ and $\omega$ yields $\zeta = |\sigma|/\sqrt{\omega^2 + \sigma^2}$. Solving for $\zeta$ for the four different values of $\tau_z$ yields 0.24, 0.26, 0.37, and 0.24, respectively. We see that the largest $\zeta$, 0.37 corresponds to the case $\tau_z = 0.1$ for which we had no overshoot and the least amount of oscillation.

MATLAB script hw5_ap5.2.m:

```matlab
%%% Problem AP5.2

t2 = [0:.01:2];
num_points2 = max(size(t2));
tauz = [0 0.005 0.1 0.5];
mp2 = []; y2 = []; tr2 = []; ts2 = []; zeros2 = []; poles2 = [];
for tauz_index = 1:max(size(tauz)),
    disp(['Open Loop:',' tauz = ',num2str(tauz(tauz_index))])
    sys1 = tf(5440*[tauz(tauz_index) 1],conv([1 0],[1 28 432]))
    disp(['Closed Loop:',' tauz = ',num2str(tauz(tauz_index))])
    sys2 = feedback(sys1,1)
y2(:,tauz_index) = step(sys2,t2); %% compute step response
    [num2,den2] = tfdata(sys2,'v'); %% get denominator of tf
    if ~(isempty(roots(num2))),
        zeros2(:,tauz_index) = roots(num2);
    else
        zeros2(:,tauz_index) = NaN; %% there's no zero
    end;
    poles2(:,tauz_index) = roots(den2); %% find poles
    ord2=max(size(den2));
    mp2(tauz_index) = max(y2(:,tauz_index));
```

yss2 = num2(ord2)/den2(ord2);  \% find steady state value of y
pmp2(tauz_index) = (mp2(tauz_index)-yss2/yss2*100;  \% find percent overshoot
r1 = 0;  r2 = 0;
for index = 1:num_points2,  \% compute 10 to 90 percent rise time
    if and(y2(index,tauz_index) > 0.1*yss2,r1==0);
        r1 = t2(index);
    end;
    if and(y2(index,tauz_index) > 0.9*yss2,r2==0);
        r2 = t2(index);
    end;
end;
tr2(tauz_index) = r2 - r1;
ts = 0;
for index = num_points2:-1:1;  \% compute 2 percent settling time
    if and(or(y2(index,tauz_index) > 1.02*yss2,...
            y2(index,tauz_index) < 0.98*yss2),ts==0);
        ts = t2(index+1);
    end;
end;
ts2(tauz_index) = ts;
end;

\%\%\% Output

disp('Values of tauz')
tauz
disp('Zeros')
zeros2
disp('Poles')
poles2
disp('Rise Times')
tr2
disp('Settling Times')
ts2
disp('Percent Overshoots')
pmp2

plot(t2,y2(:,1),t2,y2(:,2),t2,y2(:,3),t2,y2(:,4));
grid;
legend('\tau_z = 0','\tau_z = 0.005','\tau_z = 0.1','\tau_z = 0.5');
xlabel('Time (s)');  ylabel('y');  title('System Response for AP5.2');

print -depsc ap5_2.eps

Transcript of running MATLAB script \texttt{hw5\_ap5\_2.m}:

\texttt{>> hw5\_ap5\_2}
Open Loop: \( \tau_{uz} = 0 \)

Transfer function:
\[
\frac{5440}{s^3 + 28s^2 + 432s}
\]

Closed Loop: \( \tau_{uz} = 0 \)

Transfer function:
\[
\frac{5440}{s^3 + 28s^2 + 432s + 5440}
\]

Open Loop: \( \tau_{uz} = 0.005 \)

Transfer function:
\[
\frac{27.2s + 5440}{s^3 + 28s^2 + 432s}
\]

Closed Loop: \( \tau_{uz} = 0.005 \)

Transfer function:
\[
\frac{27.2s + 5440}{s^3 + 28s^2 + 459.2s + 5440}
\]

Open Loop: \( \tau_{uz} = 0.1 \)

Transfer function:
\[
\frac{544s + 5440}{s^3 + 28s^2 + 432s}
\]

Closed Loop: \( \tau_{uz} = 0.1 \)

Transfer function:
\[
\frac{544s + 5440}{s^3 + 28s^2 + 976s + 5440}
\]

Open Loop: \( \tau_{uz} = 0.5 \)

Transfer function:
\[
\frac{2720s + 5440}{s^3 + 28s^2 + 976s + 5440}
\]
\[ s^3 + 28s^2 + 432s \]

Closed Loop: \( \tau_{uz} = 0.5 \)

Transfer function:
\[
\frac{2720s + 5440}{s^3 + 28s^2 + 3152s + 5440}
\]

Values of \( \tau_{uz} \)
\[
\begin{array}{cccc}
\text{tauz} &=& 0 & 0.0050 & 0.1000 & 0.5000 \\
\end{array}
\]

Zeros
\[
\text{zeros2} =
\begin{array}{cccc}
\text{NaN} & -200 & -10 & -2 \\
\end{array}
\]

Poles
\[
\text{poles2} =
\begin{array}{cccc}
-4.0000 & -16.0000i & -4.5376 & -16.3360i & -6.5059 & -1.7514 \\
\end{array}
\]

Rise Times
\[
\text{tr2} =
\begin{array}{cccc}
0.1000 & 0.1000 & 0.0700 & 0.0300 \\
\end{array}
\]

Settling Times
\[
\text{ts2} =
\begin{array}{cccc}
0.9000 & 0.8400 & 0.4900 & 1.0600 \\
\end{array}
\]

Percent Overshoots
\[
\text{pmp2} =
\]
32.6646  28.2156  -0.0001  29.2490

Plot generated by running MATLAB script `hw5_ap5_2.m`:

![System Response for AP5.2](image)

**AP5.3** We are given a feedback system whose forward path has two or three poles and no zero. We are asked to find the percent overshoots, rise times, settling times, zeros, and closed-loop poles for values 0, 0.5, 2, and 5 of the parameter \( \tau_p \). When \( \tau_z = 0 \), the system has only two poles. Otherwise it has three.

**Solution:** The following MATLAB script achieves these objectives. The transcript and the plot of the step responses are also shown. Note also that for \( \tau_p = 0 \), the computed percentage overshoot is negative. In this case, there was no overshoot and so the percentage overshoot is undefined. The value is an artifact of the way I wrote the script. From the plot we see that adding the pole to the system decreases system stability (results in more oscillation) and increases overshoot, rise time, and settling time.

MATLAB script `hw5_ap5_3.m`:

```matlab
%%% Problem AP5.3
```
t3 = [0:.01:100];
num_points3 = max(size(t3));
taup = [0 0.5 2 5];
mp3 = []; y3 = []; tr3 = []; ts3 = []; zeros3 = []; poles3 = [];
for taup_index = 1:max(size(taup)),
    disp(["Open Loop: ", num2str(taup(taup_index))])
    sys1 = tf([1], conv([1 2 0],[taup(taup_index) 1]))
    disp(["Closed Loop: ", num2str(taup(taup_index))])
    sys3 = feedback(sys1,1)
    y3(:,taup_index) = step(sys3,t3);  %% compute step response
    [num3,den3] = tfdata(sys3,'v');  %% get denominator of tf
    disp('Poles of Closed Loop Transfer Function')
    roots(den3),  %% find poles
    ord3=max(size(den3));
    mp3(taup_index) = max(y3(:,taup_index));
    yss3 = num3(ord3)/den3(ord3);  %% find steady state value of y
    pmp3(taup_index) = (mp3(taup_index)-yss3)/yss*100;  %% find percent overshoot
    r1 = 0; r2 = 0;
    for index = 1:num_points3,  %% compute 10 to 90 percent rise time
        if and(y3(index,taup_index) > 0.1*yss3,r1==0);
            r1 = t3(index);
        end;
        if and(y3(index,taup_index) > 0.9*yss3,r2==0);
            r2 = t3(index);
        end;
    end;
    tr3(taup_index) = r2 - r1;
    ts = 0;
    for index = num_points3:-1:1;  %% compute 2 percent settling time
        if and(or(y3(index,taup_index) > 1.02*yss3,...
            y3(index,taup_index) < 0.98*yss3),ts==0);
            ts = t3(index+1);
        end;
    end;
    ts3(taup_index) = ts;
end;

%% Output

disp('Values of taup')
taup
disp('Rise Times')
tr3
disp('Settling Times')
ts3
disp('Percent Overshoots')
pmp3

plot(t3,y3(:,1),t3,y3(:,2),t3,y3(:,3),t3,y3(:,4));
grid;
legend(’\tau_p = 0’,’\tau_p = 0.5’,’\tau_p = 2’,’\tau_p = 5’);
xlabel(’Time (s)’); ylabel(’y’); title(’System Response for AP5.3’);

print -depsc ap5_3.eps

The transcript of running MATLAB script hw5_ap5_3.m follows. (I’ve taken the liberty of deleting some blank lines and some lines containing only the text “ans =” from the transcript to save space and make the transcript more readable.)

>> hw5_ap5_3
Open Loop: taup = 0

Transfer function:
    1
-----------
    s^2 + 2 s

Closed Loop: taup = 0

Transfer function:
    1
-----------
    s^2 + 2 s + 1

Poles of Closed Loop Transfer Function

   -1
   -1

Open Loop: taup = 0.5

Transfer function:
    1
---------------------
    0.5 s^3 + 2 s^2 + 2 s

Closed Loop: taup = 0.5

Transfer function:
    1
---------------------
\[ 0.5 s^3 + 2 s^2 + 2 s + 1 \]

Poles of Closed Loop Transfer Function

\[-2.8393 \]
\[-0.5804 + 0.6063i \]
\[-0.5804 - 0.6063i \]

Open Loop: \( \tau_{au} = 2 \)

Transfer function:
\[ \frac{1}{2s^3 + 5 s^2 + 2 s} \]

Closed Loop: \( \tau_{au} = 2 \)

Transfer function:
\[ \frac{1}{2s^3 + 5 s^2 + 2 s + 1} \]

Poles of Closed Loop Transfer Function

\[-2.1421 \]
\[-0.1789 + 0.4488i \]
\[-0.1789 - 0.4488i \]

Open Loop: \( \tau_{au} = 5 \)

Transfer function:
\[ \frac{1}{5s^3 + 11 s^2 + 2 s} \]

Closed Loop: \( \tau_{au} = 5 \)

Transfer function:
\[ \frac{1}{5s^3 + 11 s^2 + 2 s + 1} \]

Poles of Closed Loop Transfer Function

\[-2.0526 \]
\[-0.0737 + 0.3033i \]
-0.0737 - 0.3033i

Values of taup

taup =

0 0.5000 2.0000 5.0000

Rise Times

tr3 =

3.3500 2.6300 3.0500 4.0600

Settling Times

ts3 =

5.8400 7.5100 22.5900 53.6200

Percent Overshoots

pmp3 =

-0.0000 4.6685 27.7916 46.0542

Here is the plot generated by running MATLAB script hw5_ap5_3.m.
AP5.4 We are given the feedback system shown in Figure Ap5.4, p. 284. We are asked to find the percent overshoots, settling times, response to disturbances, and steady state errors for three values of the gain $K$. These values are 1, 10, and 100.

**Solution:** The transfer function from the reference input $R(s)$ to the output $Y(s)$ is found using the usual block diagram transformation methods from Chapter 2 to be

$$T(s) = \frac{Y(s)}{R(s)} = \frac{15K}{s^2 + 12s + (35 + 15K)}$$

(1)

and the transfer function from the disturbance input $D(s)$ to the output $Y(s)$ is likewise found to be

$$\frac{Y(s)}{D(s)} = \frac{15}{s^2 + 12s + (35 + 15K)}.$$  

(2)

We are asked to find the steady state error for a unit step input, which will be (using $E(s) = R(s) - Y(s)$ and $R(s) = 1/s$)

$$e_{ss} = \lim_{s \to 0} sE(s) = \lim_{s \to 0} s(1 - T(s))R(s) = 1 - \lim_{s \to 0} T(s) = 1 - \frac{15K}{35 + 15K} = \frac{7}{7 + 3K}.$$  

(3)

The step input and disturbance responses and requested values for each $K$ are determined using the following MATLAB script.
%%% Problem AP5.4

t4 = [0:.01:2];
num_points4 = max(size(t4));
K = [1 10 100];
for K_index = 1:max(size(K)),
    disp(['Closed Loop:',' K = ',num2str(K(K_index))])
    rtf = tf([15*K(K_index)],[1 12 (35+15*K(K_index))])
    dtf = tf([15],[1 12 (35+15*K(K_index))])
    ess(K_index) = 7/(7+3*K(K_index));
    y4r(:,K_index) = step(rtf,t4);  %% compute response to step reference input
    y4d(:,K_index) = step(dtf,t4);  %% compute response to step disturbance
    [num4,den4] = tfdata(rtf,'v');  %% get denominator of tf
    disp('Poles of Closed Loop Transfer Function')
    roots(den4),  %% find poles
    ord4=max(size(den4));
    mp4(K_index)= max(y4r(:,K_index));
    yod4(K_index) = max(y4d(:,K_index))/1;  %% unit step disturbance
    yss4 = num4(ord4)/den4(ord4);  %% find steady state value of y
    pmp4(K_index) = (mp4(K_index)-yss4)/yss4*100;  %% find percent overshoot
    ts = 0;
    for index = num_points4:-1:1,  %% compute 2 percent settling time
        if and(or(y4r(index,K_index) > 1.02*yss4,y4r(index,K_index) < 0.98*yss4),ts==0);
            ts = t4(index+1);
        end;
    end;
    ts4(K_index)= ts;
end;

%%% Output

disp('Values of K')
K
disp('Steady State Errors to Step Input')
ess
disp('Settling Times')
ts4
disp('Percent Overshoots')
pmp4
disp('max(abs(y/d))')
yod4

plot(t4,y4r(:,1),t4,y4r(:,2),t4,y4r(:,3),t4,y4d(:,1),t4,y4d(:,2),t4,y4d(:,3));
grid;
legend('Step Input, K = 1','Step Input, K = 10','Step Input, K = 100',...
'Step Disturbance, K = 1','Step Disturbance, K = 10','Step Disturbance, K = 100');
xlabel('Time (s)'); ylabel('y'); title('Step Responses for AP5.4');

print -depsc ap5_4.eps

Here's the transcript from running the MATLAB script. As before, I've taken the liberty of deleting some blank lines and some lines containing only the text “ans =” from the transcript to save space and make the transcript more readable.)

```matlab
>> hw5_ap5_4

Closed Loop: K = 1

Transfer function:
   15
-------------
s^2 + 12 s + 50

Transfer function:
   15
-------------
s^2 + 12 s + 50

Poles of Closed Loop Transfer Function

   -6.0000 + 3.7417i
   -6.0000 - 3.7417i

Closed Loop: K = 10

Transfer function:
   150
-------------
s^2 + 12 s + 185

Transfer function:
   15
-------------
s^2 + 12 s + 185

Poles of Closed Loop Transfer Function

   -6.0000 +12.2066i
   -6.0000 -12.2066i
```
Closed Loop:  \( K = 100 \)

Transfer function:
\[
\frac{1500}{s^2 + 12s + 1535}
\]

Transfer function:
\[
\frac{15}{s^2 + 12s + 1535}
\]

Poles of Closed Loop Transfer Function
\[-6.0000 + 38.7169i, -6.0000 - 38.7169i\]

Values of \( K \)
\[
K = \begin{bmatrix} 1 & 10 & 100 \end{bmatrix}
\]

Steady State Errors to Step Input
\[
ess = \begin{bmatrix} 0.7000 & 0.1892 & 0.0228 \end{bmatrix}
\]

Settling Times
\[
Ts4 = \begin{bmatrix} 0.6000 & 0.6200 & 0.6600 \end{bmatrix}
\]

Percent Overshoots
\[
pmp4 = \begin{bmatrix} 0.6488 & 21.3344 & 61.3937 \end{bmatrix}
\]

\[
\max(\abs{y/d})
\]

\[
yod4 = \begin{bmatrix} \end{bmatrix}
\]
0.3019 0.0984 0.0158

Here’s the plot. Notice that when \( K = 1 \), the responses to step inputs and step disturbances are the same, so the plot appears to show only 5 trajectories.

AP6.1 In this problem we use the Routh-Hurwitz criterion to determine the range of values of gains \( K_1 \) and \( K_2 \) for which the system having characteristic equation

\[
s^4 + 20s^3 + K_1 s^2 + 4s + K_2 = 0
\]

is stable.

Solution: The first two rows of the array are read from the coefficients of the characteristic equation. The third and fourth must be calculated. The array has the following form, where we will determine the values of \( b_3 \), \( b_1 \) and \( c_3 \).

\[
\begin{array}{cccc}
  s^4 & 1 & K_1 & K_2 \\
  s^3 & 20 & 4 & 0 \\
  s^2 & b_3 & b_1 & 0 \\
  s^1 & c_3 & 0 & 0 \\
  s^0 & d_3 & 0 & 0 \\
\end{array}
\]

(5)
We compute \( b_3 \) from
\[
b_3 = \frac{-1}{a_3} \begin{vmatrix} a_4 & a_2 \\ a_3 & a_1 \end{vmatrix} = \frac{-1}{20} (4 - 20K_1) \tag{6}
\]
and \( b_1 \) from
\[
b_1 = \frac{-1}{a_3} \begin{vmatrix} a_4 & a_0 \\ a_3 & 0 \end{vmatrix} = \frac{-1}{20} (0 - 20K_2) \tag{7}
\]
then, recalling that we can multiply a row by a positive number to simplify computations, we can simplify the computation of \( c_3 \) and \( d_3 \) by replacing \( b_3 \) and \( b_1 \) by \( \tilde{b}_3 = 20K_1 - 4 \) and \( \tilde{b}_1 = 20K_2 \). We then obtain for \( c_3 \),
\[
c_3 = \frac{-1}{b_3} \begin{vmatrix} a_3 & a_1 \\ \tilde{b}_3 & \tilde{b}_1 \end{vmatrix} = \frac{-1}{20K_1 - 4} (400K_2 + 16 - 80K_1). \tag{8}
\]
Proceeding similarly, we find \( \tilde{c}_3 = 80K_1 - 400K_2 - 16 \) and eventually \( \tilde{d}_3 = K_2 \). For stability we need all coefficients in the first column of the array to be positive (since the first element of the column is positive) so we need from row 3, \( K_1 > 1/5 \) and from row 5, \( K_2 > 0 \). Row 4 leads to the condition \( K_1/5 - 1/25 > K_2 \) so the region of stability is the set of values in the \( K_1 - K_2 \) plane below the line of slope 1/5 with \( K_1 \)-axis intercept 1/5.

**DP5.1** We are given the system shown in Figure DP5.1, p. 285, and asked to determine

(a) the closed-loop transfer function,

(b) the roots of the characteristic equation for three values of the gain \( K \), namely 0.7, 3, and 6,

(c) the expected overshoot and peak time estimated by approximating the system by a second order system corresponding to the dominant roots,

(d) the degree to which the approximation is satisfactory (by comparing the actual response with that of the second order approximation), and

(e) the gain that leads to 16

**Solution:**

(a) The closed-loop transfer function is
\[
T(s) = \frac{11.4K}{s^3 + 11.4s^2 + 14s + 11.4K}. \tag{9}
\]

(b) The roots of the characteristic polynomial for the three values of \( K \) are found in the following **MATLAB** transcript.

```
>> for K=[.7 3 6], roots([1 11.4 14 11.4*K]),end
ans =

-10.0910
-0.6545 + 0.6020i
-0.6545 - 0.6020i
```
ans =

-10.3678
-0.5161 + 1.7414i
-0.5161 - 1.7414i

ans =

-10.6889
-0.3555 + 2.5045i
-0.3555 - 2.5045i

(c) The peak time for a second order system is given by

\[ T_p = \frac{\pi}{\omega_n \sqrt{1 - \zeta^2}} \]  

so we will need to determine \( \omega_n \) and \( \zeta \) for the dominant poles. We can find these as in the first problem above using \( \zeta = \frac{\sigma}{\sqrt{\omega^2 + \sigma^2}} \) and \( \omega_n = \frac{|\sigma|}{\zeta} \). MATLAB commands that will do this are given below. Note that we must be careful when hard-coding as I have done in this example. Having checked the output of the `roots` command, I knew that the complex conjugate roots were the second and third ones and I made sure that I had the command display the results of the `roots` command so that if the order changed I would know that the result was no longer valid.

for K=[.7 3 6],
    poles = roots([1 11.4 14 11.4*K]),
    zetak = sqrt(real(poles(2))^2/(real(poles(2))^2+imag(poles(2))^2))
    omegank = -real(poles(2))/zetak
    Tp = pi/omegank/sqrt(1-zetak^2)
    po = 100*exp(-zetak*pi/sqrt(1-zetak^2))
end

A table of the results obtained in MATLAB follows.

<table>
<thead>
<tr>
<th>K</th>
<th>( \sigma )</th>
<th>( \omega )</th>
<th>( \zeta )</th>
<th>( \omega_n )</th>
<th>( T_p )</th>
<th>% Overshoot</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.7</td>
<td>-0.65</td>
<td>0.60</td>
<td>0.74</td>
<td>0.89</td>
<td>5.21</td>
<td>3.28</td>
</tr>
<tr>
<td>3</td>
<td>-0.52</td>
<td>1.74</td>
<td>0.28</td>
<td>1.81</td>
<td>1.80</td>
<td>39.41</td>
</tr>
<tr>
<td>6</td>
<td>-0.36</td>
<td>2.50</td>
<td>0.14</td>
<td>2.53</td>
<td>1.25</td>
<td>64.02</td>
</tr>
</tbody>
</table>

(d) Note that this MATLAB script again assumes the order in which the roots of the characteristic polynomial appear in the array returned by the `roots` command and must be used with caution.

%%% Problem DP5.1

clear all
t = [0:.01:10];
num_points = max(size(t));
K = [0.7 3 6];
for K_index = 1:max(size(K)),
    disp(['Closed Loop:',' K = ',num2str(K(K_index))])
    sys = tf(11.4*[K(K_index)],[1 11.4 14 11.4*K(K_index)])
    poles = roots([1 11.4 14 11.4*K(K_index)]),
    asys = tf([-11.4*K(K_index)/poles(1)],conv([1 -poles(2)],[1 -poles(3)]))
    ysys(:,K_index) = step(sys,t);   %% compute step response
    yasys(:,K_index) = step(asys,t); %% compute step response
end;

%% Output
plot(t,ysys(:,1),t,ysys(:,2),t,ysys(:,3),...
     t,yasys(:,1),t,yasys(:,2),t,yasys(:,3));
grid;
legend('K= 0.7','K = 3','K = 6','as: K = 0.7','as: K = 3','as: K = 6');
xlabel('Time (s)'); ylabel('y');
title('Step Response for DP5.1: System and Approximation (as)');
print -depsc dp5_1.eps

The script above generates the following transcript.

>> hw5_dp5_1d
Closed Loop:   K = 0.7

Transfer function:
    7.98
---------------------
s^3 + 11.4 s^2 + 14 s + 7.98

poles =

-10.0910
-0.6545 + 0.6020i
-0.6545 - 0.6020i

Transfer function:
    0.7908
---------------------
s^2 + 1.309 s + 0.7908
Closed Loop:  $K = 3$

Transfer function:

\[
\frac{34.2}{s^3 + 11.4s^2 + 14s + 34.2}
\]

poles =

-10.3678
-0.5161 + 1.7414i
-0.5161 - 1.7414i

Transfer function:

\[
\frac{3.299}{s^2 + 1.032s + 3.299}
\]

Closed Loop:  $K = 6$

Transfer function:

\[
\frac{68.4}{s^3 + 11.4s^2 + 14s + 68.4}
\]

poles =

-10.6889
-0.3555 + 2.5045i
-0.3555 - 2.5045i

Transfer function:

\[
\frac{6.399}{s^2 + 0.7111s + 6.399}
\]

The plot indicates that the approximation is rather good.
(d) Finally, we will use the approximation given by the dominant roots to obtain a value for $K$ that results in a 16% overshoot. I did this by writing a MATLAB script to test values of $K$ between 0.7 and 3 since the percent overshoots for those values of $K$, 3.28% and 39.41% respectively, bracket 16%. (I did this by replacing “for $K = [0.7 \ 3 \ 6]$” by “for $K = [0.7:0.1:3]$” in the for loop used to calculate $T_p$ and the percent overshoot in part (c).) With $K = 1.3$, I obtained percent overshoot of 15.46 and $T_p = 3.03$ seconds.

**DP6.3** In this problem we are given a system with unity negative feedback and forward path containing

$$G(s) = \frac{K(s + 2)}{s(1 + \tau s)(1 + 2s)}$$

and asked to determine the regions of stability for the system, then choose values for $K$ and $\tau$ that result in steady-state error to a ramp input less than or equal to 25 percent of the input magnitude, and finally to determine the percent overshoot resulting from this choice of parameters.

**Solution:** The closed-loop transfer function is

$$T(s) = \frac{K(s + 2)}{2\tau s^3 + (2 + \tau)s^2 + (K + 1)s + 2K}$$
so the Routh-Hurwitz array is

\[
\begin{array}{ccc}
  s^3 & 2\tau & K + 1 \\
  s^2 & 2 + \tau & 2K \\
  s^1 & (2 + \tau)(K + 1) - 4\tau K & 0 \\
  s^0 & 2K & 0 \\
\end{array}
\]  \quad (13)

where we have multiplied the third and fourth rows by appropriate constants to simplify the computations. We will require all elements of the first column to be positive. (We could repeat the calculations for negative values if we wished.) From row 1 we need \( \tau > 0 \); from row 2, \( \tau > -2 \), from row 4, \( K > 0 \), and from row 3, \( 3\tau K < 2K + 2 + \tau \). We could solve for a constraint on \( \tau \) in terms of \( K \) or vice versa. We choose the latter and obtain the constraint

\[
K < \frac{2 + \tau}{3\tau - 2}.  \quad (14)
\]

From this we see that \( K > 0 \) requires \( \tau > 2/3 \). Plotting upper bound on \( K \) vs. \( \tau \) yields the following plot, where as noted, the stability region is to the right of \( \tau = 2/3 \) and below the curve.

![Stable region below curve and to right of \( \tau = 2/3 \) for DP6.3](image)
MP6.2 We are asked to determine the poles (roots of the denominator) and zeros (roots of the numerator) of the closed loop system having

\[ G(S) = \frac{K(s^2 - s + 2)}{s^2 + 2s + 1} \]  \hspace{1cm} (15)

in the forward path and negative unity feedback for three values of \( K \). We are then asked about the stability of the closed-loop system in each case. The MATLAB script used to generate the roots follows.

\[
K = [1 2 5];
for \text{index} = 1:max(size(K)),
    \text{display(['Open Loop, } K = ',\text{num2str(K(index))})
    olsys = tf(K(index)*[1 -1 2],[1 2 1])
    display(['Closed Loop, } K = ',\text{num2str(K(index))})
    clsys = feedback(olsys,1)
    [num,den]=tfdata(clsys,'v')
    display('Zeros')
    \text{roots(num)}
    display('Poles')
    \text{roots(den)}
end;
\]

The transcript below shows that the closed-loop poles corresponding to \( K = 1 \) both have negative real part so in this case the closed-loop system is stable. On the other hand, when \( K = 5 \), the closed-loop poles have positive real part and the closed-loop system is unstable. For \( K = 2 \), the poles have zero real part and the system is marginally stable. (I’ve taken the liberty of deleting some blank lines and some lines containing only the text “\text{ans} =” from the transcript to save space and make the transcript more readable.)

\[
\text{>> hw5_mp6_2}
\]

\text{Open Loop, } K = 1

\text{Transfer function:}
\[ s^2 - s + 2 \]
\[ \frac{}{s^2 + 2s + 1} \]

\text{Closed Loop, } K = 1

\text{Transfer function:}
\[ s^2 - s + 2 \]
\[ \frac{}{2s^2 + s + 3} \]

\text{Zeros}
0.5000 + 1.3229i  
0.5000 - 1.3229i

Poles

-0.2500 + 1.1990i  
-0.2500 - 1.1990i

Open Loop, $K = 2$

Transfer function:
\[
\frac{2 s^2 - 2 s + 4}{s^2 + 2 s + 1}
\]

Closed Loop, $K = 2$

Transfer function:
\[
\frac{2 s^2 - 2 s + 4}{3 s^2 + 5}
\]

Zeros

0.5000 + 1.3229i  
0.5000 - 1.3229i

Poles

0 + 1.2910i  
0 - 1.2910i

Open Loop, $K = 5$

Transfer function:
\[
\frac{5 s^2 - 5 s + 10}{s^2 + 2 s + 1}
\]

Closed Loop, $K = 5$

Transfer function:
\[
\frac{5s^2 - 5s + 10}{6s^2 - 3s + 11}
\]

Zeros
\[0.5000 + 1.3229i\]
\[0.5000 - 1.3229i\]

Poles
\[0.2500 + 1.3307i\]
\[0.2500 - 1.3307i\]